Group members: Eduard Koshkelyan, Marwane Bidamou, Roshan Maharjan

Question 1.

Write an algorithm beautiful(A, n)
Input : An integer array with n elements such that the best-case running time is equal to the worst-case running time. Write the algorithm and give your analysis to justify your claim.

```java
public static int beautiful(int[] A, int n){
    int sum = 0;
    for (int i = 0; i < n; i++) {
        sum += A[i];
    }
    return sum;
}
```

Explanation: The best-case and worst-case running times of the beautiful function are the same because the function always iterates through all n elements of the array, regardless of the array's content. For every element, the function performs a constant-time operation (adding it to the sum), meaning the time complexity is O(n) in both cases. There are no conditions that would cause early termination or skipping any elements, ensuring that the algorithm consistently processes the entire array.

Question 2

Order them based on their complexity.
2^n , 2^(2n), 2^(n + 1), 2^(2^n)

Ans: 2^n, 2^(n + 1), 2^(2n), 2^(2^n)

Question 3

Mention one algorithm you know for each of the time complexities listed.
Ans:
O(1) = Accessing Array Index
O(log n) = Binary Search
O(n) = Looping through an array
O(n log n) = Merge Sort
$O(n^2)$ = Bubble Sort
$O(n^3)$ = Using 3 loops to find all possible combinations of 3 element sets of n elements.

Group members: Eduard Koshkelyan, Marwane Bidamou, Roshan Maharjan

$O(2^n)$ = Using recursion to find fibonacci number of n

Question 4

Apply Master Theorem and determine the time complexity of fib(n) shown in Lesson 2. If you cannot apply Master Theorem please give detailed explanation.

Ans: Master Theorem cannot be used for fib(n) shown in lesson 2 because in order to use this formula there needs to be recurrences that arise from Divide-And-Conquer algorithms. For fib(n) in lesson 2 it is just decreasing numbers by 1 and 2 for each recursive call. For the Master formula to work it needs to divide n elements to equal parts.

$$T(n) = \begin{cases} d & \text{if } n = 1 \\ aT(\lceil \frac{n}{b} \rceil) + cn^k & \text{otherwise} \end{cases}$$

b in the above formula represents a number to equally divided parts and it does not work for Fibonacci recursive functions.